AF/2857
LFW

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants:	Mark J. Kraffert	§	Group Art Unit:	2857
		§		
Serial No.:	09/776,364	§		
		§	Examiner:	Jeffrey R. West
Filed:	February 2, 2001	§		
		§		
For:	Sharing Data Files in a Test Environment	§	Atty. Dkt. No.:	MCT.0134US (MUEI-0558.00/US)

Mail Stop Appeal Brief – Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

APPEAL BRIEF TRANSMITTAL

Sir:

Transmitted herewith in triplicate is the Appeal Brief in this application with respect to the Notice of Appeal filed on June 14, 2004.

Copies of the same Appeal Brief were submitted on the same day without a signed certificate of mailing. Please disregard the Appeal Brief mailed without the signed certificate of mailing, and consider instead the attached Appeal Brief with signed certificate of mailing. Please apply the payment of $330.00 sent in the separate mailing to the Appeal Brief enclosed herewith.

The Commissioner is hereby authorized to charge any fees that may be required, and/or credit any overpayment to Deposit Account No. 20-1504. A duplicate copy of this sheet is enclosed.

08/19/2004 HGUTEMA1 00000040 201504 09776364
01 FC:1402 330.00 DA

Date of Deposit: August 16, 2004
I hereby certify under 37 CFR 1.8(a) that this correspondence is being deposited with the United States Postal Service as **first class mail** with sufficient postage on the date indicated above and is addressed to the Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450.

Ginger Yount
Ginger Yount

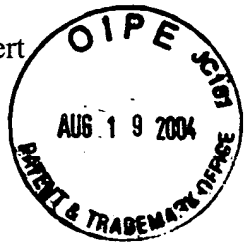
Respectfully submitted,

Date: 8-16-04

Dan C. Hu, Reg. No. 40,025
TROP, PRUNER & HU, P.C.
8554 Katy Freeway, Suite 100
Houston, TX 77024
713/468-8880 [Telephone]
713/468-8883 [Facsimile]

CUSTOMER NO. 21906

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE



Applicants:	Mark J. Kraffert	Group Art Unit:	2857
Serial No.:	09/776,364		
Filed:	February 2, 2001	Examiner:	Jeffrey R. West
For:	Sharing Data Files in a Test Environment	Atty. Dkt. No.:	MCT.0134US (MUEI-0558.00/US)

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF

Sir:

Applicant respectfully appeals from the final rejection mailed March 11, 2004.

I. REAL PARTY IN INTEREST

The real party in interest is Micron Technology, Inc., by virtue of the assignment recorded at Reel/Frame 012391/0370.

II. RELATED APPEALS AND INTERFERENCES

None.

III. STATUS OF THE CLAIMS

Claims 1-14, 17-21, and 23-32 have been finally rejected and are the subject of this appeal. Claims 15, 16, and 22 were previously canceled.

IV. STATUS OF AMENDMENTS

No amendments were submitted after final rejection.

08/19/2004 HGUTEMA1 00000040 09776364

01 FC:1402 330.00 DA

Date of Deposit: August 16, 2004
I hereby certify under 37 CFR 1.8(a) that this correspondence is being deposited with the United States Postal Service as **first class mail** with sufficient postage on the date indicated above and is addressed to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313.

Ginger Yount

Ginger Yount

V. SUMMARY OF THE INVENTION

In a typical test environment, there may be several functional areas where tests are performed. For example, a manufacturing company may have the following functional areas: order entry, factory planning, manufacturing, shipping, and invoice/accounting. In performing tests in each of the functional areas, it may sometimes be desirable to use the same data files. In many instances, the data files from one test may have to be physically copied to a location that is accessible by a test system in the next test. This is generally inefficient and is often associated with errors, since a wrong file may be copied. Specification, p. 1, lines 13-19.

Fig. 1 of the specification shows a test environment 10 that includes a network 16 coupled to a plurality of test systems 12, 14. Although only two test systems are illustrated, more than two can be used in further examples. Specification, p. 2, lines 20-23.

As further illustrated in Fig. 1, two databases 18 and 20 are also coupled to the network 16. A first database 18 is a TEST database, while a second database 20 is a DEVL or development database. An operator at one of the test systems 12, 14 can select one of the databases 18, 20 to perform a test on. The designation of TEST and DEVL for the databases is provided by way of example only, as further embodiments may employ other types of databases. Specification, p. 2, lines 24-29.

For a test performed by each test system 12, 14, one or more data files are used. The data files contain data that are used for performing data-driven tests on the database 18 or 20. In accordance with some embodiments, a common set of data files 24 are shared by the different tests systems 12, 14. Thus, for example, the test system 12 can use the data files 24 in a first test, and the test system 14 can use the same set of data files in the next test. In accordance with some embodiments, each of the test systems 12, 14 includes a data source routine 124 (Fig. 2) that provides a convenient mechanism for the different test systems 12, 14 to share the same data

files 24. By using the data source routine 124, manual manipulation of the test systems 12, 14 to use the same data files 24 can be avoided. By receiving certain parameters, a data source routine 124 is able to identify the name of a data file to use. If in each test system the same parameter values are received, then the data source routine 124 will provide the same file name for identifying the data file to use in each test. Specification, p. 2, line 30-p. 3, line 11.

Fig. 2. shows components of the test system 12 or 14. The test system 12 or 14 includes a test module 122 that is able to perform tests of the database system 18 or 20 over the network 16. In one example embodiment, the test module 122 is the WINRUNNER testing tool from Mercury Interactive Corporation. In other embodiments, other types of testing tools can be used in the test system 12 or 14. Specification, p. 3, lines 12-16.

In performing tests, the test module 122 establishes a communications session with the database system 18 or 20 over the network 16. The communication session is established between a communications client 116 in the test system 12 or 14 and a communication server (not shown) in the database system 18 or 20. In one embodiment, the communications session is a Telnet session, which involves terminal emulation over a network. In terminal emulation, a client (the test system 12 or 14) behaves as though it is a terminal of another computer (the host), which in the example of Fig. 2 is the database system 18 or 20. Thus, in this example, the communications client 116 is a Telnet client. Once a Telnet session is established, the test system 12 or 14 is able to access files and software in the database system 18 or 20. In other embodiments, other types of communications sessions are possible over the network 16 between the test system 12 or 14 and the database system 18 or 20. Specification, p. 3, lines 17-28.

Also, as mentioned above, the test system 12 or 14 includes the data source routine 124 that enables the identification of a common set of data files 24 that can be shared by multiple test

systems. The data source routine 124 can be invoked by the test module 122 for identifying the name of one of the data files 24. Although shown as separate components, the test module 122 and data source routine 124 may be part of the same integrated software module. For example, the data source routine can be a subroutine, function, or object that can be invoked within the test module 122. Specification, p. 4, lines 2-8.

The data source routine 124 is capable of identifying file name of a default data file 100 or a common file 102 stored in the storage unit 22. There may be plural common files, with one file for each of the TEST and DEVL database system 18 or 20. The default data file 100 and common files 102 make up the common set of data files. As shown in Fig. 2, the storage unit 22 may be located in a server 110 (e.g., a network server or some other system accessible over the network 16). Specification, p. 4, lines 16-21.

Referring to Fig. 3, in accordance with one example, the common set of data files 24 are shared by test systems 200, 202, 204, 206, and 208 in different functional areas. For example, the test system 200 is used to test an order entry area, the test system 202 is used to test a factory planning area, the test system 204 is used to test a manufacturing area, the test system 206 is used to test a shipping area, and a test system 208 is used to test an invoice/accounting area. The data source routine 124 executable in each of the test systems 200, 202, 204, 206, and 208 will automatically identify the correct data file to use without manual configuration of each test system or the creation of different custom scripts in each test system to find the correct file name. In some embodiments, all that needs to occur is the provision of predetermined parameters to the data source routine 124 (either by the test module 122 or by the user) to enable the identification of the file name of the common data file. Specification, p. 4, line 22-p. 5, line 2.

Referring to Fig. 4, a process according to one embodiment performed by the data source routine 124 is illustrated. The data source routine 124 is called by the test module 122. In the call, the test module 122 can set a Clarify parameter and a DBase parameter. The Clarify parameter can have a predefined common value, with the Clarify parameter value being part of the file name of a common file 102. The DBase parameter specifies the database to be tested, either the TEST database 18 or the DEVL database 20. Specification, p. 5, lines 3-8.

The data source routine 124 determines (at 302) whether a Clarify parameter was received in the call from the test module 122. If not, the data source routine 124 then prompts (at 304) the user for a Clarify parameter value. Next, the data source routine 124 determines if the user has entered a Clarify parameter value (at 306). The user is given a predetermined period of time to enter the Clarify parameter value. If a Clarify parameter value was not received, then the data source routine 124 sets a parameter DataFile equal to DefaultDataFile (at 308). The value DefaultDataFile refers to the default data file 100 (Fig. 2). Next, the data source routine 124 sets (at 310) a parameter DBNum to the value 2, which corresponds to the DEVL database 20. Thus, if the Clarify parameter is not received, then the default data file 100 is used and the DEVL database 20 is tested. Specification, p. 5, lines 9-18.

If the data source routine 124 determines (at 302 or 306) that the Clarify parameter has been received, then the data source routine 124 next determines if the DBase parameter is received (at 312). If not, the data source routine 124 prompts the user (at 314) for the DBase value (either DEVL or TEST). The data source routine 124 waits (at 316) for entry of the database name by the user. If a database name is not entered after a predetermined time period, the parameter DataFile is set to the DefaultDataFile value and DBNum is set to the value 2 (at 308 and 310). Specification, p. 5, lines 19-25.

If the database name has been received (at 312 or 316), then the value of DBNum is set to the appropriate value (1 for TEST or 2 for DEVL). Next, a parameter FString is set (at 320) to the concatenation of the Clarify and DBase parameters, that is,

FString = Clarify, DBase.

The data source routine 124 then performs (at 322) a directory list command on the server 110 in which the common set of data files 24 are kept. The output of the directory command is routed to a file FILE.TXT. The file FILE.TXT is then opened and a search is performed to find file names that contain the string FString. Specification, p. 5, line 26-p. 6, line 6.

Next, it is determined if an error occurred (at 326). An error may occur if a match is not found or if there are more than one file name containing the string FString. If an error is determined, then an error code is returned (at 328) and the DataFile and DBNum parameters are set at 308 and 310. Specification, p. 6, lines 5-8.

However, if an error did not occur (at 326), then the parameter DataFile is set to the matching file name. This is the file that is then used as the data file for the test to be performed by the test module 122. Specification, p. 6, lines 9-11.

VI. ISSUES

A. Are Claims 1, 2, And 5-13 Obvious Over Slutz In View Of Fujimori?

B. Are Claims 3, 4, 14, 17-19, 23, 24, And 27-32 Obvious Over Slutz In View of Fujimori And Talley?

C. Are Claims 1-14, 17-19, 23, 24, And 27-32 Obvious Over Gartner In View Of Fitting?

D. Are Claims 20, 21, 25, And 26 Obvious Over Either (1) Slutz, Fujimori, Talley, And Walls; or (2) Gartner, Fitting, And Walls?

VII. GROUPING OF THE CLAIMS

Group 1: Claims 1, 2

Group 2: Claims 6-11

Group 3: Claims 14, 18, 19

Group 4: Claims 23, 24, 32

Group 5: Claims 3, 4, 17, 27, 29-31

Group 6: Claims 12, 13

Group 7: Claims 20, 21, 25, 26

Within each group, the claims stand and fall together. Claims 5 and 28 are not part of any group.

VIII. ARGUMENT

All claims should be allowed for the reasons set forth below.

A. Are Claims 1, 2, And 5-13 Obvious Over Slutz In View Of Fujimori?

Independent claim 1 recites a method of performing a test that comprises performing a first test with a first test system, performing a second test with a second test system, and in each of the first and second test systems, receiving plural parameters, and in each of the first and second test systems, identifying a file name of a first data file to use in each of the first and second tests based on the plural parameters. Further, the method includes the first and second test systems using the first data file in performing the respective first and second tests.

As conceded by the Examiner, Slutz does not teach identifying a file name of a first data file to use based on received plural parameters. However, contrary to the assertion in the Examiner, Slutz also fails to disclose the following element of claim 1: the first and second test systems using *the first* data file in performing the respective first and second tests. Thus, what is

performed by a method according to claim 1 is that the two test systems that perform respective first and second tests both use the same data file, which in claim 1 is the first data file. This is clearly not taught or suggested by Slutz.

In asserting that Slutz teaches that first and second test systems use *the same* data file in performing respective first and second tests, the Examiner focused on column 5, lines 32-33, which states that the test program "reads in" a configuration file 400 containing a set of parameters for a test procedure. *See* 3/11/2004 Office Action at 11. The Examiner also cited to column 4, lines 11-20, of Slutz, which states that a medium 134 stores instructions and data for the test program. *See* 3/11/2004 Office Action at 11. The column 5 passage cited by the Examiner refers to tasks performed by a test program 300 that resides in *one of the PC clients 120*. Slutz, 5:1-2. Alternatively, Slutz mentions that the test program 300 can execute within server 130, or at any other convenient location. Slutz, 5:3-4. In other words, the discussion in column 5 of Slutz focuses on *one* test program running on *one* system (a PC client, a server, or other location). Thus, the teaching that the test program reads in a configuration file 400 in column 5 refers to the reading of a configuration file by *one* test program residing on one system.

In column 4, Slutz does mention that "the test program can be executed in one or more of clients 120 or even in server 130." Slutz, 4:18-20. However, this teaching must be construed in view of the remaining teachings of Slutz. Slutz focuses on a test program to generate database query language statements that are syntactically correct according to a query language. By use of a configuration file, a test program can generate the desired set of statements. One of the parameters in the configuration file 400 specifies the name of a database 260. Slutz, 5:33-35. The testing process is not limited to one or more fixed databases for testing a DBBMS, but rather can employ arbitrary, user-selected target databases. Slutz, 5:35-37. To select different target

databases, different configuration files would have to be used. It would hardly seem efficient for multiple PC clients of Slutz to run test programs each accessing the *same* configuration file 400--that would result in the *same* test being performed by the test programs in multiple client PCs. Thus it is clear that Slutz does not provide any specific teaching or suggestion that multiple test systems can use the same data file to perform first and second tests.

In the Advisory Action dated June 24, 2004, the Examiner again attempted to pick separate passages from Slutz as providing the teaching that first and second test systems use a first data file in performing respective first and second tests. The Examiner specifically pointed to preliminary blocks 310 (Fig. 3 of Slutz) that begin an entire testing process, as well as block 311 for reading a configuration file 400. Note that blocks 310 and 311 are performed by *a test generator 300* (Slutz, 4:66). There is no indication whatsoever in the acts performed in blocks 310 and 311 (of Fig. 3 of Slutz) that multiple clients would use the same configuration file. The statement at lines 18 and 19 in column 4 of Slutz that a test program can be executed in one or more clients 120 does not provide any teaching that the test programs in these clients would access this same configuration file. This conclusion by the Examiner is based on speculation that is inconsistent with the actual teachings of Slutz.

In view of the foregoing, even if Slutz can be properly combined with Fujimori, the hypothetical combination does not teach or suggest each and every element of claim 1. Therefore, a *prima facie* case of obviousness has not been established for at least this reason.

Furthermore, there was no motivation or suggestion to combine Slutz and Fujimori in the manner proposed by the Examiner. The reliance on Fujimori as suggesting a modification of Slutz to achieve the claimed invention is misplaced. Fujimori relates to an electronic musical instrument having a memory for storing musical tone information containing waveform data and

assigning a file name to the file that stores the waveform data. Fujimori, Abstract. As discussed in column 5 of Fujimori, character strings can be entered to form a file name. However, Applicant notes that Fujimori has nothing to do with identifying a file name of a data file to use in *first and second tests* based on plural parameters. All Fujimori would have suggested to a person of ordinary skill is a technique for assigning a file name for storing musical tone information. Such a person of ordinary skill in the art would not have been motivated by the teaching of Fujimori to identify a file name of a data file to use, by first and second test systems, in first and second tests based on plural parameters.

It is well established law that "[t]he mere fact that the prior art could be so modified would not have made the modification **obvious** unless the prior art suggested the **desirability** of the modification. *In re Gordon*, 733 F.2d 900, 902, 221 U.S.P.Q. 1125 (Fed. Cir. 1984) (emphasis added). As the Federal Circuit has stated, "virtually all [inventions] are combinations of old elements." *In re Rouffet*, 149 F.3d 1350, 1357, 47 U.S.P.Q.2d 1453 (Fed. Cir. 1998). "Most, if not all, inventions are combinations and mostly of old elements." *Id.* "Therefore an examiner may often find every element of a claimed invention in the prior art. If identification of each claimed element in the prior art were sufficient to negate patentability, very few patents would ever issue. Furthermore, rejecting patents solely by finding prior art corollaries for the claimed elements would permit an examiner to sue the claimed invention itself as a blueprint for piecing together elements in the prior art to defeat the patentability of the claimed invention. Such an approach would be 'an illogical and inappropriate process by which to determine patentability.'" *Id.*

A person looking to the teachings of Fujimori would have learned that a file name for sampling waveform data can be created by manually inputting a first portion (FNB1) and

combining with another portion FNB2. Fujimori, Fig. 5, 5:49-6:8. This teaching of Fujimori would not have suggested identifying a file name of a first data file *to use in each of the first and second tests* based on received plural parameters. All Fujimori teaches to a person of ordinary skill in the art is that the file name of musical data can be found by combining two different character strings.

There was absolutely no suggestion whatsoever of any desirability of incorporating a file name combination technique for musical data, as taught by Fujimori, into the database test environment of Slutz. The Examiner stated that Fujimori was cited only to teach combining two strings/parameters to form a file name. However, the actual teachings of Fujimori cannot be ignored to ascertain whether a person of ordinary skill in the art would have been motivated to combine the teachings of Slutz and Fujimori. Here, the Examiner ignored the actual teachings of Fujimori in arguing that Slutz and Fujimori can be combined to achieve the claimed invention. That is clearly not the case. A person of ordinary skill in the art looking to the teachings of Slutz and Fujimori would clearly have not been led to the claimed invention. Therefore, because no motivation or suggestion existed to combine Fujimori and Slutz, a *prima facie* case of obviousness with respect to claim 1 has not been established for this further reason.

Claim 5, which depends from claim 1, further recites that each of the tests is performed on a database, and where one of the parameters represents the database. Note that "one of the parameters" referred to in claim 5 is one of the plural parameters that are used to identify a file name of the first data file. There is no indication in Slutz or Fujimori that a parameter that can be used in conjunction with at least another parameter to identify file name represents a database. In Fujimori, the character string FNB1 is manually input (Fujimori, 6:18), and the character string FNB2 represents the name of a note that is written into a register RFNB2 (Fujimori,

7:29-33). FNB1 and FNB2 are combined to identify a name of a file. Fujimori, 6:6-8. However, neither FNB1 nor FNB2 represent a "database," as recited in claim 5.

With respect to independent claim 6, it is respectfully submitted that there was no motivation or suggestion to combine Slutz and Fujimori to combine first and second values to generate a file name of a test file to use in a test, for the reasons given above. Therefore, a *prima facie* case of obviousness has not been established with respect to the claim.

Moreover, with respect to dependent claim 12 (which depends indirectly from claim 6), Slutz also fails to teach or suggest performing a second test in a second system using *the* test file. As discussed above, Slutz refers to reading different configuration files by test programs in PC clients in performing its tests. Thus, in Slutz, two systems do not perform a test using the same test file, as recited in claim 12.

For the foregoing reasons, it is respectfully requested that the final rejection of claims 1, 2, and 5-13 over Slutz and Fujimori be reversed.

B. Are Claims 3, 4, 14, 17-19, 23, 24, And 27-32 Obvious Over Slutz In View of Fujimori And Talley?

Independent claim 14 recites a test system having an interface to a network coupled to a storage unit containing a data file for use in a test, a control unit, and a routine executable on the control unit to receive a first parameter and a second parameter and to combine the first and second parameters to form a string, the second parameter representing a database to perform a test on. The routine identifies a file name of the data file based on the string, and a test module is executable on the control unit to perform the test using the data file.

With respect to claim 14, the Examiner stated that the combination of Slutz and Fujimori "does not specifically disclose searching for the data file in storage for use in testing a database." 3/11/2004 Office Action at 4. Claim 14 does not actually recite such language. The Examiner

does not really address why reference is made to the language "searching for the data file in storage for use in testing a database" with respect to claim 14.

Similar to arguments presented above in connection with claim 1, the hypothetical combination of Slutz and Fujimori does not teach or suggest combining first and second parameters to form a string, and a routine to identify a file name of a data file (for use in a test) based on the string.

Talley also does not teach or suggest a routine to identify a file name of a data file based on a string that is formed from the combination of received first and second parameters. Although Talley discusses searching for a configuration file, it discusses this in the context of searching for the configuration file in a current directory or in a user's home directory. Talley does not teach or suggest identifying a file name of a data file based on a string that is formed from the combination of received first and second parameters. Therefore, even if the combination of Slutz, Fujimori, and Talley are proper, the hypothetical combination fails to teach or suggest the invention of claim 14. A *prima facie* case of obviousness has thus not been established with respect to claim 14 for at least this reason.

Moreover, there is no motivation or suggestion to combine the teachings of Slutz, Fujimori, and Talley. As noted above, Fujimori is directed to a teaching that is completely unrelated to forming a string from plural parameters to identify a file name of a data file to use in a test. Talley also does not provide any teaching or suggestion of combining parameters to form a string for the purpose of identifying a file name of a data file to use in a test. Therefore, there was no motivation or suggestion to combine the teachings of Slutz, Fujimori, and Talley. The *prima facie* case of obviousness fails for this further reason.

Independent claim 23 was also rejected over the asserted combination of Slutz, Fujimori, and Talley. As conceded by the Examiner, Slutz and Fujimori fails to disclose searching a predetermined directory on a device to find a test file containing the string. However, contrary to the Examiner's assertion, Talley does not disclose searching a predetermined directory on a device to find a test file containing a string that is concatenated from received first and second parameters. Talley describes a database interface to access a database configuration file that contains descriptions of contents of each of the databases. Talley, 1:54-57. The configuration file describes the characteristics and contents of a particular database. Talley, 4:17-20. Talley describes operations that look for a configuration file and determines if the configuration file exists in the user's home directory. Talley, 6:18-23. However, looking for the configuration file of Talley is not the same as searching a directory to find a *test* file, as recited in the claim. Therefore, even if the references can be combined, the hypothetical combination of Slutz, Fujimori, and Talley fails to disclose or suggest each and every element of claim 23. Moreover, as discussed above, no motivation or suggestion existed to combine the teachings of Slutz, Fujimori, and Talley. The *prima facie* case of obviousness against claim 23 is therefore defective.

Independent claim 27 was also rejected over the asserted combination of Slutz, Fujimori, and Talley. Contrary to the Examiner's assertion, Talley fails to disclose or suggest a routine executable to search a directory to find a file name of one of data files that contains a string concatenated from first and second parameters to use for a test. Therefore, the hypothetical combination of Slutz, Fujimori, and Talley fails to teach or suggest each and every element of claim 27. Also no motivation or suggestion existed to combine the teachings of the three references. The *prima facie* case of obviousness against claim 27 is therefore also defective.

Independent claim 28 was also rejected over the combination of Slutz, Fujimori, and Talley. Contrary to the Examiner's assertion, there is no teaching or suggestion in Talley of searching a predetermined directory on a device to find a *test* file containing a string that is concatenated from a common parameter. Therefore, the hypothetical combination of Slutz, Fujimori, and Talley fails to teach or suggest each and every element of claim 28. Furthermore, there is no motivation or suggestion to combine the references. The *prima facie* case of obviousness against claim 28 is therefore defective.

For the foregoing reasons, it is respectfully requested that the final rejection of claims 3, 4, 14, 17-19, 23, 24, and 27-32 over Slutz, Fujimori, and Talley be reversed

C. Are Claims 1-14, 17-19, 23, 24, And 27-32 Obvious Over Gartner In View Of Fitting?

Claim 1 was also rejected as being obvious over the asserted combination of Gartner and Fitting. The asserted combination of references, even if proper, does not teach or suggest all elements of claim 1. As discussed above, claim 1 recites performing a first test with a first test system; performing a second test with a second test system; in each of the first and second test systems, receiving plural parameters; identifying a file name of a first data file to use in each of the first and second tests based on the plural parameters; and using the first data file in performing the respective one of the first and second tests.

Gartner relates to a system and method for testing a database system that includes external file references to files stored on a remote file system. The database system stores a control table that enables access to files on the remote file system accessed through the external file references. Gartner, Abstract. The database system provides for definitions of relations that accommodate the existence of an attribute that is according to an external file reference (EFR) data type. Gartner, 5:42-46. The data structure of the EFR data type includes the name of a

server and the name of a file (file name). Gartner, 5:46-47. The EFR data type supports database system behavior that causes the database system to issue a "LinkFile" command to an appropriate file server for the named file when an enterprise user issues an SQL call. Gartner, 5:55-59. A table that contains an attribute according to the EFR data type is table 60, shown in Fig. 2 of Gartner. A query can be made to retrieve contents of the table 60 in the database system. The query returns results that include one or more servers/filename references. Gartner, 6:19-24.

Nowhere within the teaching of Gartner is there any indication of *first and second test systems* that (1) receive plural parameters, (2) identify a file name of a first data file to use in the first and second tests, and (3) use the first data file in performing the respective first and second tests. Gartner discloses retrieving an external file reference from a table stored in a database system, with the file name identifying a file in an external file system. The identified file is retrieved from the external file system and used to test the database system. Gartner, 3:5-20.

The Examiner stated that the plurality of users and plurality of applications for testing the database are considered to be the first and second test systems. 3/11/2004 Office Action at 7. This assertion ignores the express language of claim 1, which recites several roles for the first and second test systems. The applications and users in Gartner clearly do not receive plural parameters, do not identify a file name of a first data file to use in the first and second tests based on the plural parameters, and do not use the first data file in performing respective first and second tests. Therefore, for at least this reason, a *prima facie* case of obviousness has not been established over Gartner and Fitting and is defective.

Thus, contrary to the Examiner's assertion, Gartner does not teach "many of the features in the claimed invention." Fitting does not supply the teaching necessary to combine with

Gartner to achieve the claimed combination. In Fitting, an empty file is generated by a test system to send to a database server. The database server, from this empty file, generates a database query to obtain a model number. This model number is provided back to the requesting test system by appending the model number to the file name of the empty file. In this way, the products that are being tested do not need to store their model information, since the test systems are able to retrieve the model information based on the identifier of each product. Thus, there is nothing in Fitting to even remotely suggest that first and second test systems use the same data file for performing respective first and second tests. Since neither Gartner nor Fitting teaches or suggests the claimed invention, their hypothetical combination also does not teach or suggest the claimed invention.

The Examiner cited to column 5, lines 15-23, of Fitting as disclosing "the invention for sharing files between a plurality of test systems, each able to executive [sic] a plurality of different tests." 3/11/2004 Office Action at 8. The cited column 5 passage of Fitting refers to configuring and controlling test equipment 116 to test product 118. The cited passage also states that the controller of the test system has a plurality of test routines, one for each of different models of the product. However, there is no mention in this passage of sharing files. A *prima facie* case of obviousness has thus not been established with respect to claim 1 over Gartner and Fitting.

In addition, no motivation or suggestion existed to combine the teachings of Gartner and Fitting. Whereas Gartner relates to a testing method and apparatus for evaluating the performance of a database system that includes external file references, Fitting describes a quality control system that uses bi-directional messaging that includes empty files. In the technique of Fitting, a request file having the format <product identifier> <requested data type>

is sent to a shared file directory 127. Fitting, 4:16-26. In response to this request, a return file is created by the shared file directory 127, where the return file is a renamed version of the request file. Fitting, 4:50-53; 6:39-43. There is absolutely no reason or desirability to incorporate the quality control system of Fitting into the database test system of Gartner. Therefore, because no motivation or suggestion existed to combine the teachings of Gartner and Fitting, the *prima facie* case of obviousness fails for this further reason.

Independent claim 6 is also allowable over the asserted combination of Gartner and Fitting. As conceded by the Examiner, Gartner does not specifically disclose combining first and second parameters to form a file name. 3/11/2004 Office Action at 7. Applicant further submits that Gartner also does not disclose or suggest receiving a second value representing a database to perform a test on. The Examiner cited to column 5, lines 41-54, of Gartner for the teaching of receiving the second value representing a database to perform a test on. A closer review of Gartner will reveal that the cited passage actually refers to storing an external file reference within a table, such as table 60, in a database management system. The external file reference refers to an external file system containing test files. These external file references do not constitute the second value representing a database to perform a test on. The test files in the external file system are actually files used during testing of the database system. Therefore, Gartner is lacking a number of items asserted to be disclosed by Gartner in the Office Action.

In response to the above arguments, the Examiner stated that the external file references are the databases being tested. 3/11/2004 Office Action at 17. This statement contradicts the teaching of Gartner itself, which shows the database system being tested as being DBMS 15. The external file reference refers to test files -- they are not the databases being tested.

Fitting also fails to disclose the missing elements. Fitting does not teach or suggest receiving a second value representing a database to perform a test on, in conjunction with combining the first value and the second value to generate a file name of a test file to use in a test. Thus, since neither Gartner nor Fitting teaches or suggest the claimed invention, their hypothetical combination also does not teach or suggest the claimed invention. A *prima facie* case of obviousness has thus not been established with respect to claim 1.

Independent claims 14, 23, 27, and 28 are allowable over the asserted combination of Gartner and Fitting for similar reasons.

For the foregoing reasons, it is respectfully requested that the final rejection of claims 1-14, 17-19, 23, 24, and 27-32 over Gartner and Fitting be reversed.

D. Are Claims 20, 21, 25, And 26 Obvious Over Either (1) Slutz, Fujimori, Talley, And Walls; or (2) Gartner, Fitting, And Walls?

In view of the defective rejections of base claims over the asserted combination of (1) Slutz, Fujimori, and Talley or (2) Gartner and Fitting, it is respectfully submitted that the rejections of dependent claims over (1) Slutz, Fujimori, Talley, and Walls, or (2) Gartner, Fitting, and Walls, are also defective.

Therefore, reversal of the final rejections of claims 20, 21, 25, and 26 over (1) Slutz, Fujimori, Talley and Walls; and (2) Gartner, Fitting, and Walls, is respectfully requested.

IX. CONCLUSION

It is respectfully submitted that all final rejections be reversed and the claims be allowed to issue.

Respectfully submitted,

Date: 8-16-04

Dan C. Hu, Reg. No. 40,025
TROP, PRUNER & HU, P.C.
8554 Katy Fwy, Ste 100
Houston, TX 77024-1805
713/468-8880 [Phone]
713/468-8883 [Facsimile]

CLAIMS ON APPEAL

1. A method of performing a test, comprising:

performing a first test with a first test system;

performing a second test with a second test system:

in each of the first and second test systems, receiving plural parameters;

in each of the first and second test systems, identifying a file name of a first data file to use in each of the first and second tests based on the plural parameters; and

the first and second test systems using the first data file in performing the respective first and second tests.

2. The method of claim 1, further comprising performing at least another test with at least another test system using the first data file.

3. The method of claim 1, further comprising, in each of the first and second test systems, accessing a storage system over a network to find a file name containing strings in each of the plural parameters.

4. The method of claim 3, wherein accessing the storage system comprises accessing the storage system to find a file name containing a concatenation of the strings.

5. The method of claim 1, wherein each of the tests is performed on a database, and wherein one of the parameters represents the database.

6. A method of performing a test, comprising:

receiving a first value;

receiving a second value representing a database to perform a test on; and

combining the first value and the second value to generate a file name of a test file to use in the test.

7. The method of claim 6, wherein receiving the first value comprises receiving a predetermined string, the predetermined string being part of the file name of the test file.

8. The method of claim 6, further comprising performing the test using a test module and invoking a routine, from the test module, to generate the file name of the test file.

9. The method of claim 8, further comprising executing the test module in a test system.

10. The method of claim 9, further comprising the test module performing a test on the database coupled over a network.

11. The method of claim 6, further comprising performing the test using a first test system, wherein the receiving and combining acts are performed in the first test system.

12. The method of claim 11, further comprising, in a second system:

receiving the first value;

receiving the second value representing the database;

combining the first value and the second value to generate the file name of the test file; and

performing another test on the database using the test file.

13. The method of claim 12, wherein the first test system performs a first type of test and the second test system performs a second type of test.

14. A test system comprising:

an interface to a network coupled to a storage unit containing a data file for use in a test;

a control unit;

a routine executable on the control unit to receive a first parameter and a second parameter and to combine the first and second parameters to form a string, the second parameter representing a database to perform a test on,

the routine to identify a file name of the data file based on the string; and

a test module executable on the control unit to perform the test using the data file.

17. The test system of claim 14, wherein the routine is executable to access the storage unit and to search file names on the storage unit for a file name containing the string.

18. The test system of claim 14, wherein the test module is executable on the control unit to perform a test of the database coupled to the network.

19. The test system of claim 18, wherein the test module is executable to pass the first and second parameters to the routine.

20. The test system of claim 19, wherein the routine is executable to prompt a user for one or both of the first and second parameters if not passed by the test module.

21. The test system of claim 20, wherein the routine is executable to set a file name of a default data file if not received from the test module or the user.

23. A method of performing a test, comprising:

receiving a first parameter containing a predetermined value;

receiving a second parameter representing a database to perform a test on;

concatenating the first parameter and the second parameter to generate a string that is at least a portion of a file name; and

searching a predetermined directory on a device to find a test file containing the string.

24. The method of claim 23, further comprising accessing the device over a network to search the predetermined directory.

25. The method of claim 23, further comprising:

prompting a user for a value of the first parameter; and

setting a default value for the first parameter if the first parameter value is not received from the user.

26. The method of claim 25, further comprising:

prompting the user for a value of the second parameter; and

setting a default value for the second parameter if the second parameter value is not received from the user.

27. A system comprising:

an interface to a network coupled to a storage unit containing a directory of data files;

a control unit;

a routine executable on the control unit to receive a first parameter and a second parameter and to concatenate the first and second parameters to form a string, the first parameter containing a predetermined value, and the second parameter representing a database to perform a test on,

the routine executable to search the directory to find a file name of one of the data files that contains the string and to set the one data file as the data file to use for the test; and

a test module executable on the control unit to perform the test.

28. A method of performing tests, comprising:

receiving a predetermined common parameter;

receiving a second parameter representing a database to perform a test on;

concatenating the common parameter and the second parameter to generate a string that is at least a portion of a file name; and

searching a predetermined directory on a device to find a test file containing the string,

wherein receiving the common parameter, receiving the second parameter, concatenating the common parameter and the second parameter, and searching the predetermined directory is performed in each of plural test systems.

29. The method of claim 1, further comprising:

combining the plural parameters to form a string; and

locating the first data file by finding the file name containing the string.

30. The method of claim 6, further comprising locating the test file having the file name.

31. The test system of claim 14, the routine to locate the data file by finding the file name containing the string.

32. The method of claim 23, wherein searching the predetermined directory comprises searching the predetermined directory to find the test file having a name containing the string.